Sample dispersion is better than sample discrepancy for classification

Benoît Gandar [1] Gaëlle Loosli [2] Guillaume Deffuant [3]

Research Report LIMOS/RR-03-17

October 1, 2010

1. LISC, CEMAGREF, LIMOS- Campus des cézeaux - 63173 Aubière
2. LIMOS - Campus des cézeaux - 63173 Aubière
3. LISC, CEMAGREF - Campus des cézeaux - 63173 Aubière

Abstract

We want to generate learning data within the context of active learning. First, we recall theoretical results proposing discrepancy as a criterion for generating sample in regression. We show surprisingly that theoretical results about low discrepancy sequences in regression problems are not adequate for classification problems. Secondly we propose dispersion as a criterion for generating data. Then, we present numerical experiments which have a good degree of adequacy with theory.

Keywords: Active Learning, Statistical Learning, Space Filling Design, Discrepancy, Dispersion.

Résumé

Notre objectif est la génération de données dans le contexte de l'apprentissage actif. Tout d'abord, nous rappelons des résultats théoriques proposant la discrépance comme critère pour la génération d'échantillons en régression. Nous montrons que ces résultats théoriques sur la dispersion faible ne sont pas adéquats pour le cas de la classification. Ensuite nous proposons le critère de dispersion pour la génération de données et présentons des résultats numériques qui illustrent les résultats théoriques.

Mots clés : Apprentissage actif, apprentissage statistique, répartition dans l'espace, discrépance, dispersion

Acknowledgement / Remerciements

This research was supported by the European project PATRES (NEST-043268).

1 Introduction

We consider a problem of active learning: how to generate a finite relevant learning data when user is able to generate data himself. This problem can arise in various contexts such as optimization of a meta-model in engineering (see Jourdan and Zabalza-Mezghani, 2004), function approximation (see Chapel and Deffuant, submitted) or kernel approximation in theory of viability (see Deffuant et al., 2007) for example. This problem of space filling design also arise when we want to model an interaction between variables without prior knowledge and data come from experiments such as ground taking, chemical experiments or biology. It's well known that the more numerous the data are, the best quality the modeling is. However obtaining data can be very expensive or destructive in consequence the experimenter can only proceed to a fixed number of experiments. He has to choose the best learning set without prior knowledge about results of analysis or experiments. Mathematically, it is like supposing that we can determine, with an oracle, the label of any point in a given set, and we want generate a sample of a given size where the points are chosen a priori independently of the target function and independently of the learner in order to get the best approximation of the oracle function.

Determining the best learning set for active learning of functions (regression) is an already solved problem. Cervellera and Muselli (2004) showed that de-randomizing improves convergence rates for smooth target functions when compared to naive random samplers. Using results on approximation of integrals, they and Cervellera et al. (2008) show theoretically that low discrepancy samples give the best results for a regression problem or density estimation. They provide an empirical demonstration of these results in the specific case of the multi-layer perceptron. Mary (2005) refines this theoretical demonstration in his PhD thesis.

We show that the theoretical approach to obtain generalization error bounds in regression is not adapted to classification. This result is somehow surprising, because classification can be seen as a particular case of functions approximation.

An analysis in depth suggests that dispersion (or covering radius: the radius of the higher ball containing no points) is probably a pertinent indicator of quality for samples to be used in active classification. Indeed, using a simple classification algorithm (as nearest neighbors), we establish a theoretical link between generalization error and dispersion.

We present experimental results using K-nearest neighbors (KNN) and SVMs (see Schlkpf and Smola, 2002) that confirm this hypothesis.

In the second part of this document, we explain notion of discrepancy of sequence, we present theoretical results on active regression, and we show that, surprisingly, these results cannot be transferred to classification (or manifold boundaries learning). In the third part, we show with theoretical arguments that minimizing the dispersion of the sample is the relevant strategy to insure the best results in active classification learning problems while limiting the sample size. In the fourth part we present numerical experiments. In the last part, we discuss about these results and conclude .

2 Results about active regression learning do not apply to active classification learning

We recall mathematical context of learning and present notion of discrepancy of a sequence. Then, we present error bounds in regression using discrepancy. The last part shows that results about regression cannot be apply to classification.

2.1 Framework and results about learning

Suppose we have $\{(x_1, f(x_1)), (x_2, f(x_2)), \ldots, (x_n, f(x_n))\}$, a set of labeled examples. The objective of learning is to approximate as precisely as possible the function f, by a function $\hat{f}$, obtained with a learning algorithm and a set D_n. The case of function f taking its values in $\mathbb{R}$ refers to regression or function learning, the case of f taking its values in a finite set refers to classification or manifold boundaries learning . In this paper we consider only the binary classification setting: other classifications can be viewed as a extensions of binary classification. In many practical situations, the set D_n cannot be freely chosen: the observations x_i are viewed as realizations of random variable. Learning theory is also a stochastic theory and is called statistical learning. However in several cases, positions of the points x_i can be selected in the space by choice of experiments or simulations. This framework is called deterministic or active learning.

The goal is hence to find $\hat{f}$, an approximation of function f, in a family of functions Γ. We can evaluate punctually the quality of function $\hat{f} \in \Gamma$ at each point x by a loss function l which measures the difference between the output of $\hat{f}$ and the function f. The loss function l must be symmetric, semi-positive and $l(y_1, y_2) = 0$ if and only if $y_1 = y_2$. The principal loss functions used are $l(y, \hat{f}(x)) = \left|y - \hat{f}(x)\right|^p, p \in \mathbb{N}^*$ in the case of regression, and $l(y, \hat{f}(x)) = \begin{cases} 0 \text{ if } y = \hat{f}(x) \\ 1 \text{ if } y \neq \hat{f}(x) \end{cases}$ in the case of classification.

Global quality of function $\hat{f}$ is defined by the expected risk or generalization error:

$$R(\hat{f}) = \int l(f(x), \hat{f}(x)).$$

A typical way of choosing the best function $\hat{f} \in \Gamma$ consists in choosing function which minimizes the expected risk, itself depending on the unknown target function f. Unfortunately, in this learning framework, we cannot compute it. The pioneer work by Vapnik and Chervonenkis (1971) investigates the procedure called Empirical Risk Minimization (ERM). It consists in selecting the best function regarding the training data. The empirical risk of function $\hat{f} \in \Gamma$ is defined by:

$$\hat{R}_n(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} l(\hat{f}(x_i), y_i).$$

It is a stochastic estimation of the expected risk $R(\hat{f})$ which depends on the stochastic learning set D_n. The most natural way to select a function $\hat{f} \in \Gamma$ is to choose the function which minimizes this empirical risk on the first n examples of learning.

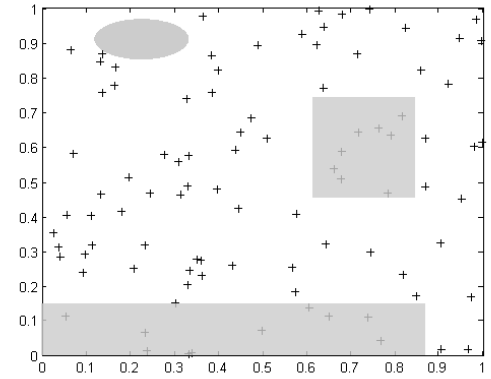
2.2 Discrepancy of a sequence

Discrepancy is a measure of spread of a sequence widely employed in numerical analysis as in Niederreiter (1992), Drmota and Tichy (1997) or Morokoff and Caflisch (1995). In this part, we present the mathematical definition of discrepancy, its proprieties and how to generate sequences with low discrepancy.

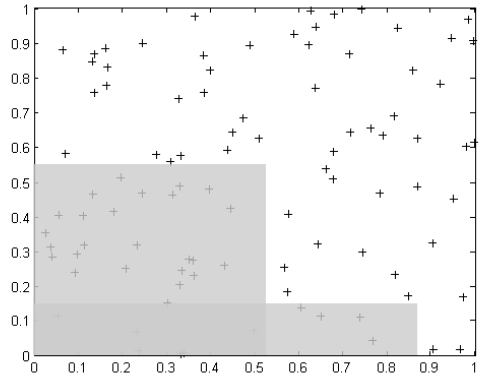
Discrepancy of a sequence: Discrepancy of a sequence can be viewed as a quantitative measure for good "uniformity" of a sequence, or its spreading. Considering without loss of generality that our sample has to be taken inside the unit hypercube I^s, of dimension s, discrepancy is the maximal difference on all the convex subsets of I^s between the proportion of the points in the convex subset and the volume of the convex subset (see FIG. 1(a)). Niederreiter (1992) shows that this definition is closely related to the star discrepancy, in which, instead of considering any convex subsets of I^s, we only consider hyper-rectangles containing origin (see FIG. 1(b)).

More formally, we note I^{s*} the set of all subintervals of I^s of the form $\Pi_{i=1}^{s}[0, u_i)$, λ the Lebesgue measure and $\#$ the operator which, for a sequence $x = x_{(n)} = \{x_1, \ldots, x_n\}$ with n elements and a set P, gives the number of element of x in the set P. We consider only the star discrepancy $D_n^*(x)$ of an n-sequence x defined by (see FIG. 1(b)):

$$D_n^*(x) = D^*(x_{(n)}) = \sup_{P \in I^{s*}} \left| \frac{\#(P, c_{(n)})}{n} - \lambda(P) \right|.$$



(a) Estimation of discrepancy with convex



(b) Estimation of discrepancy with hyper-rectangle containing origin: star discrepancy $D_n^*(x)$

Figure 1: Methods to estimate dispersion

How to estimate discrepancy: Estimation of discrepancy is not very easy. Niederreiter (1992) has developed a formula to compute it numerically in the case where $s = 1$ or $s = 2$. Moreover Thiedmard (2001) has developed an algorithm to estimate star discrepancy. However its complexity is exponential so it's barely useable.

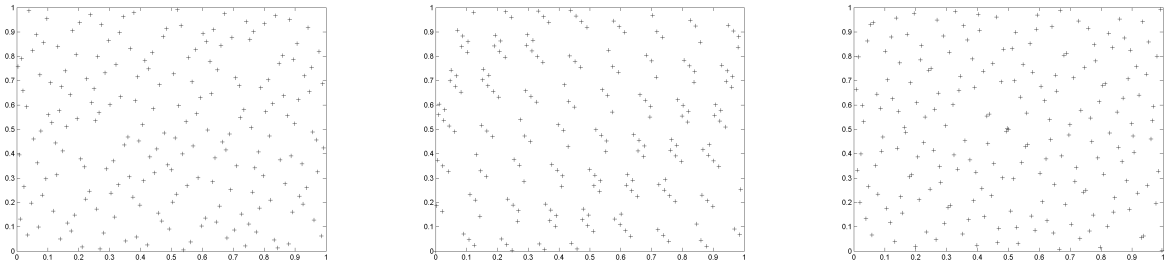
Discrepancy of uniform random variables: Let $U_1, \ldots, U_n$ a sequence of uniform random variable obtained with the uniform law on I^s. Denoting $F_n(t) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{U_i \in [0,t]}$ the empirical repartition function associated to the n first variables and

$$D_n^*(x) = D_n^*(U_1, \ldots, U_n) = \sup_{t \in I^s} \{|F_n(t) - \Pi_{i=1}^s t_i|\},$$

$$\text{then: } \limsup_{n \to +\infty} \sqrt{\frac{2n}{ln(ln(n))}} D_n^*(x) = 1.$$

Low discrepancy sequence: A low discrepancy sequence is defined as a sequence which its star discrepancy is asymptotically better than star discrepancy of a uniform random sequence. Nowadays, best discrepancies of (infinite) low discrepancy sequences known are asymptotically $O\left(\frac{log^s(n)}{n}\right)$, and $O\left(\frac{log^{s-1}(n)}{n}\right)$ for a finite sequence of size n. According to Niederreiter (1992), discrepancy of an uniform random sequence is about $O\left(\frac{1}{\sqrt{n}}\right)$ in mean. Note that a n-regular grid has also a discrepancy of order $O\left(\frac{1}{\sqrt{n}}\right)$, which is not low. Graphically, low dispersion sequence can be characterized by the fact that values of her projection on each axis are more numerous than for a regular grid.

How to generate low discrepancy sequence: It exists many algorithms to generate low discrepancy sequences. Usual low discrepancy sequences used for integral estimation are Halton sequences, Faure sequences, Sobol sequences or Hammersley sequences. Three of these sequences are represented on figure 2. In this paper, we don't talk about their implementation and we invite the reader to see for instance Niederreiter (1992) or Tezuka (1995).



(a) Sequence of Halton. (b) Sequence of Faure. (c) Sequence of Sobol.

Figure 2: Three low discrepancy sequences of 200 points.

2.3 Error bounds in regression with low discrepancy sequence

Cervellera and Muselli (2004) apply the theorem of Koksma-Hlawka, which bounds integral approximation, in the context of statistical learning of regression. They ob-

tain that for any function g in the family function Γ:

$$\begin{aligned} \left| R(g) - \hat{R}_n(g) \right| &= \left| \int l(y, g(x)) - \frac{1}{n} \sum_{i=1}^{n} l(g(x_i), y_i) \right| && (1) \\ &\leq V_{HK}\left(l(f,g)\right) D_n^*(x) && (2) \\ &\leq V_{HK}\left(|g-f|^p\right) D_n^*(x) && (3) \end{aligned}$$

In equations 2 and 3, V_{HK} is a particular measure of variation of a function: the variation of Hardy-Krause. For more detail about it, see appendix A. It is interesting to note that this equation permits to deal separately with the issues of model complexity and sample complexity. When this variation is bounded for the function $|g-f|^p$, it is possible to bound up the deterministic generalization error, with a upper bound in $O(D_n^*)$. Using low discrepancy sequence implicates the upper bound $O\left(\frac{log^s(n)}{n}\right)$. It proves the convergence of $\hat{R}_n(g)$ to $\inf_{g\in\Gamma} R(g)$ with the principle (ERM) when all functions $\{l(f,g) : g \in \Gamma\}$ are uniformly bounded (see Cervellera and Muselli, 2004).
In his PhD-thesis, Mary (2005) proves that it doesn't need to bound uniformly functions $\{l(f,g) : g \in \Gamma\} = \{|g-f|^p : g \in \Gamma\}$. We only have to suppose that variation of the target function $V_{HK}(f)$ and variation of function $\arg\min_{g\in\Gamma} R(g)$ are finite: therefore we can also consider bigger functions family. Within the context of statistical learning (see Vapnick, 1995), the empirical estimation of a function decreases as $O\left(\frac{1}{\sqrt{n}}\right)$ and with a fixed confidence level. It is a convergence in probability. When using low discrepancy sequence, we obtain a deterministic upper bound that decreases as $O\left(\frac{log^s(n)}{n}\right)$. This speed is significantly quicker when the dimension s is small.

Numerical experiments with regression: Cervellera and Muselli (2003) and Mary (2005) have made experiments: they have then experimentally proved that reducing discrepancy is a relevant strategy in the case of learning functions.

2.4 Results cannot be transferred to classification

Classification can be seen as a particular case of functions approximation. So, theoretically, we can transpose results from regression to classification.

Comparison with Vapnik-Chervonenkis ($\mathcal{VC}$) bounds: In equations 2 and 3, variation function plays the same role as the $\mathcal{VC}$ dimension in statistical learning theory literature, and consequently the condition to be in finite $\mathcal{VC}$ dimension is substituted by an hypothesis of finite variation of functions. Supposing this hypothesis is right, it is so not necessary to have a null empirical risk estimation of target function to obtain an upper bound of the error that decreases as $O\left(\frac{1}{n}\right)$instead of

$O\left(\frac{1}{\sqrt{n}}\right)$.
For empirical risk minimization Vapnick (1995) considers a sequence of functions which have a finite and increasing $\mathcal{VC}$-dimension and obtain a stochastic convergence with speed about $O\left(\sqrt{\frac{log(n)}{n}}\right)$. When using low discrepancy sequence, we always obtain a deterministic convergence of about $O\left(\frac{log^s(n)}{n}\right)$.

Results cannot be theoretically transferred to classification: Classification is the case where function f takes its values in the set $\{0, 1\}$. The associated loss function is also an indicator function. However the variation of indicator functions is generally infinite [4] (see Owen, 2004). So the superior bound in inequality 3 is equal to infinity too. Previous results cannot also be theoretically transposed to this case.

Numerical experiments comfort this result: We have made numerical tests in order to illustrate this hypothesis with classification problems. The experimental protocol and results are presented in section 4.2. To summarize them, we don't obtain better results with low discrepancy sequence than with a regular grid (which have a higher discrepancy). Therefore discrepancy does not seem to be the relevant criterion to get optimal samples for classification. Moreover, Morokoff and Caflisch (1995); Press and Teukolsky (1989) have demonstrated that to estimate integrals, using low discrepancy sequences is not efficient when dimension is large or when integrand function is not much smooth or has discontinuities. It is typically the case of classification problem when integrand function is an indicator function. As a consequence, it consolidates our hypothesis.

3 Low dispersion is a better criterion of sample quality for active classification learning

Previous results are taken a leaf out of discrepancy and more generally of multidimensional integration. But discrepancy is not the one and only possible criterion to characterize uniformity of a sequence. Dispersion, or covering radius, is another estimator of spread of a sequence used in numerical optimization. It is usually used in iterative algorithms in order to approximate the extremum of a non derivable function in a compact set. The error approximation can also be theoretically expressed by a function of dispersion (see Niederreiter (1992)). Contrary to discrepancy, dispersion isn't a measure but is a criterion based on distance. Now, we consider unit cube I^s with the euclidian distance d.

4. Only indicator functions which have discontinuities depending on axes have a finite variation.

Dispersion of a sequence: Dispersion of a sequence $x = \{x_1, \ldots, x_n\}$ is defined by:

$$\delta(x) = \sup_{y \in I^s} \min_{i=1,\ldots,n} d(y, x_i)$$

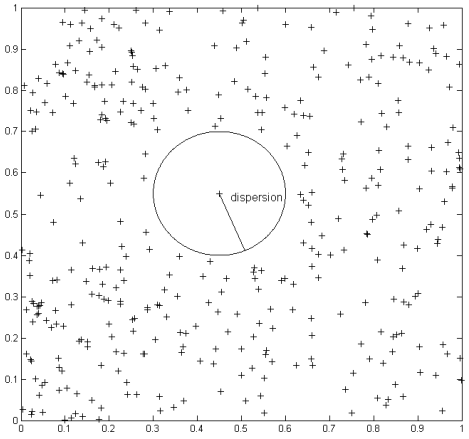


Figure 3: Estimation of dispersion

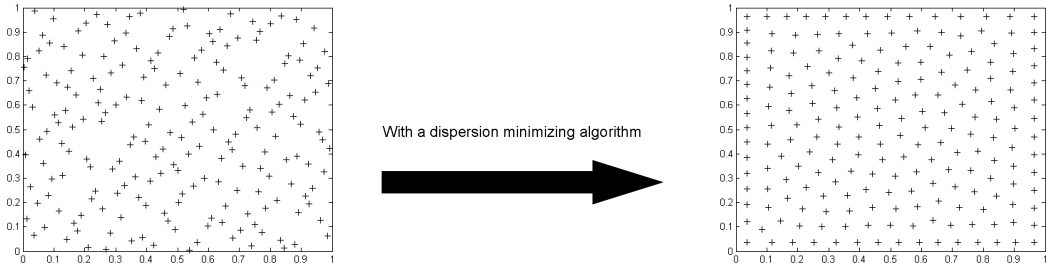
Intuitively dispersion of a sequence is the radius of the biggest empty ball of I^s and Figure 3 shows it graphically .

Remarks about discrepancy and dispersion: Discrepancy and dispersion are not equivalent measures. Indeed, when we add a point in a sequence, its dispersion can only decrease [5]. Its discrepancy can increase or decrease. Moreover, for an appropriate number of point, the configuration which minimizes the dispersion is a regular grid which doesn't minimize discrepancy.
To illustrate these differences, we have represented in dimension 2 in figure 4(a) a low discrepancy Halton sequence with 200 points (and a measured dispersion of 0,010). With an algorithm described in section 4.3 we have displaced the points in order to decrease dispersion. Figure 4(b) shows the final result: dispersion is equal to 0,002. Note on this figure that the points tend to form a regular grid, which does not have a low discrepancy.

Generalization error is function of dispersion for a simple classification learning: The purpose of this paragraph is to established a link between the generalization error and the dispersion of the learning set using a learning process similar to the nearest neighbors.

Theorem: Let f a function from I^s to $\{-1, +1\}$. We want to approximate it with a learning set E of dispersion δ. Denoting $B(x, R)$ the ball of center x and radius R.

5. At worst, it is equal.



(a) Halton low discrepancy sequence: dispersion = 0,010.

(b) Previous sequence modified: dispersion = 0,002.

Figure 4: Two sequences of 200 points.

Let $\chi_{f^+} = \{x \in I^s | f(x) = +1\}$ and $\chi_{f^-} = \{x \in I^s | f(x) = -1\}$, sets of antecedent values of function. We suppose f has this property of regularity: $\exists R$ such

- $\forall x \in \chi_{f^+}, \exists x_0 \in \chi_{f^+} | x \in B(x_0, R)$ and $B(x_0, R) \subset \chi_{f^+}$
- $\forall x \in \chi_{f^-}, \exists x_0 \in \chi_{f^-} | x \in B(x_0, R)$ and $B(x_0, R) \subset \chi_{f^-}$

Let the learning algorithm A approximating the function f by $A(E) = \hat{f}$ as :

$$\hat{f}(x) = \begin{cases} +1 & \text{si } \forall x_i^- \in E \cap \chi_{f^-}, d(x_i^-, x) \geq 2\delta. \\ -1 & \text{si } \forall x_i^+ \in E \cap \chi_{f^+}, d(x_i^+, x) \geq 2\delta. \\ \text{random} & \text{otherwise} \end{cases}$$

There is $\lambda > 0$ such as, for any learning set E of dispersion $\delta < R$, the algorithm A gives an approximation of f with a generalization error $L(A(E))$ such as: $L(A(E)) < \lambda\delta$.

Proof: Let: $F^+ = \{x \in I^s | \forall x_i^- \in E \cap \chi_{f^-}, d(x_i^-, x) \geq 2\delta\}$ and $F^- = \{x \in I^s | \forall x_i^+ \in E \cap \chi_{f^+}, d(x_i^+, x) \geq 2\delta\}$.

1. Prove that $F^+ \subset \chi_{f^+}$. Let $x \in F^+$. Supposing $x \in \chi_{f^-}$. Regularity hypothesis of f implies: $\exists x' \in \chi_{f^-} | x \in B(x', R)$ et $B(x', R) \subset \chi_{f^-}$. All the more $\exists x'' \in \chi_{f^-} | x \in B(x'', \delta)$ and $B(x'', \delta) \subset \chi_{f^-}$, because $R > \delta$. By definition of the dispersion, $\exists x_0 \in E$, such as $x_0 \in B(x'', \delta)$. Therefore $d(x, x_0) < 2\delta$, it is in contradiction to the hypothesis ($x \in F^+$). So $x \in \chi_{f^+}$. The learning algorithm does not make mistakes on F^+. Hence we have $F^- \subset \chi_{f^-}$.

2. Estimation of generalization error: $L(A(E)) = \int_{I^s} \left| f - \hat{f} \right| (x)dx$. On F^+ and F^-, f et $\hat{f}$ are equal, therefore errors are in the set $I^s - F^+ - F^-$, which distings F^+ from F^-. So $L(A(E)) = \int_{I^s - F^+ - F^-} \left| f - \hat{f} \right| (x)dx$. So $L(A(E)) < V(I^s - F^+ - F^-)$, where V is the volume of this set. Let ∂f the boundary between χ_{f^-} and χ_{f^+}, et $M = \{x \in I^s | d(x, \partial f) \leq 2\delta\}$.
It's evident that $I^s - F^+ - F^- \subset M$. Indeed, $x \notin F^+$ implicates $d(x, E \cap \chi_{f^-}) < 2\delta$, that implicates $d(x, \chi_{f^-}) < 2\delta$. Hence, we demonstrate that $d(x, \chi_{f^+}) < 2\delta$. Therefore $L(A(E)) < V(M)$. But $V(M) \leq 4\delta S(\partial f) \left(\frac{R+2\delta}{R}\right)^{s-1} \leq 4\delta S(\partial f) 3^{s-1}$,

where $S(\partial f)$ is the integral on the surface ∂f. Regularity condition on f insures that this integral is finite. This factor defined with R allows to bound the volume, supposing the radius of curvature of ∂f is at its minimal value R.

Conclusion: In this particular algorithm, the generalization error is directly linked to the dispersion of the learning set. We can think, with this result, that dispersion is a pertinent indicator to measure the quality of a learning set in classification.

4 Numerical experiments

In this section, we present some numerical experiments using low discrepancy and low dispersion sequences. Firstly, we present the protocol used to generate learning problems. Secondly, we present results showing that discrepancy is not the relevant criterion to generate sample for classification problem. Then, we present a new algorithm to generate low dispersion sample. The last part shows that minimizing the dispersion is a relevant strategy for classification learning problems.

4.1 Examples of generated classification problems

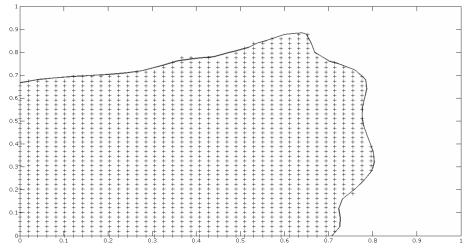
We have generated two types of classification problems for dimension 2 to 8. The first type is relative to simple classification problems. These problems are very common in nature or in studies of model or meta-model. Classification boundaries have small variations and are smooth. Moreover they are nearing the hypothesis of theorem presented in section 3. Graphical representations of this type of classification rules are presented in dimension 2 at figure 5(a) & figure 5(b), and in dimension 3 at figure 5(c) & figure 5(d).

The second type of problems is relative to difficult classification problems. Classification boundaries have more important variations and are less smooth. Moreover classifications surfaces have more connex components. Graphical representations of this type of classification rules are presented in dimension 2 at Fig.6(a) & Fig.6(b), and in dimension 3 at Fig.6(c) to Fig.6(f).

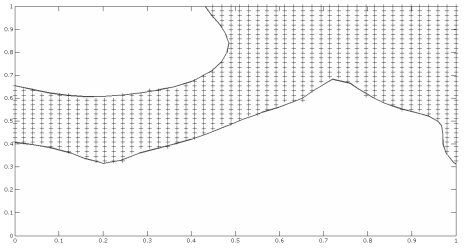
4.2 Experiments about discrepancy

In this section, we present experimental results about learning with low discrepancy sequences and regular grid, and compare them.

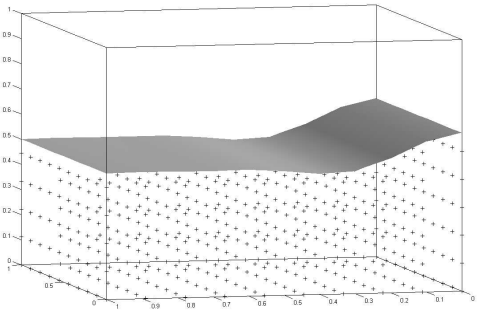
We have generated set of classification problems of first type. For each problem, we have made learning with three different samples of same size and two learning methods. The first learning sample is a regular grid, the other two samples are the most classical low discrepancy sequences: the Halton sequence for the second sample (noted LDS1) and the Sobol sequence for the third (noted LDS2). Halton and Sobol sequence are two different sequences which have a well known discrepancy of about $O\left(\frac{log^{s-1}(n)}{n}\right)$ and are based on different algorithms for their generation.
For learning algorithms, we use KNN and SVM with these configurations:



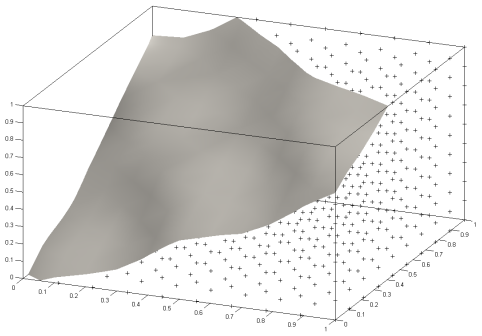
(a) Example 1 in dimension 2



(b) Example 2 in dimension 2



(c) Example 1 in dimension 3



(d) Example 2 in dimension 3

Figure 5: Examples of simple learning problems in dimension two and three

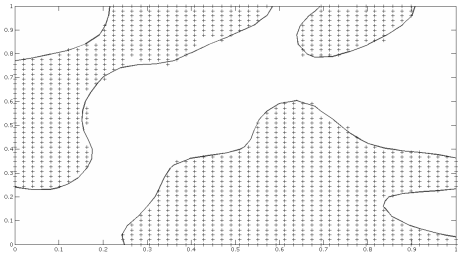
– k=3 for KNN.
– SVM : we have made an 5-folds-cross-validation error (for each problem) to choose the rbf kernel parameter σ between 0.2 to 0.8.
– Parameter C is fixed to 10000 [6].
– we have worked with toolbox libSVM developed by Chang and Lin (2001).

Table 4.2 shows average values of generalization error for 1000 simple learning problems. The error is estimated on a regular sequence of 6000 points.

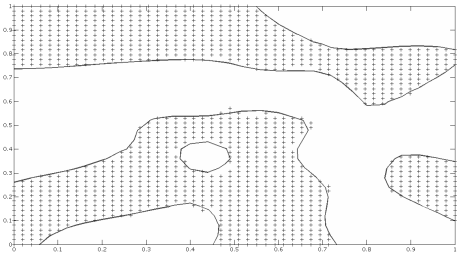
For each set of problems and for each learning method, we have also tested the statistical relevance of the results. We used a Student statistical test at the 5% significance level. Samples with a significantly lower mean generalization error have been ranked 1. In the same way, we have allocated rank 3 to samples with a statistically upper average. If two different averages are not statistically different, we have allocated the same rank.

We can remark that generally the average on a regular grid is lower than the average on low discrepancy sequences (LDS1 and LDS2). As a consequence, it is clear that low discrepancy is not a relevant criterion to generate samples for classification problems. Moreover the discrepancy effect is more observable on KNN than on SVM.

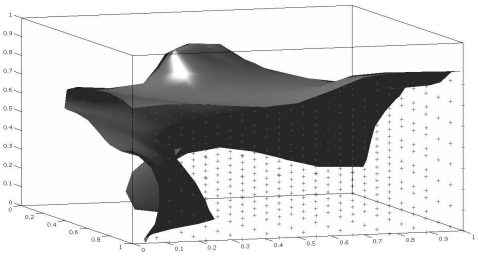
6. Our problems are hard-margin.



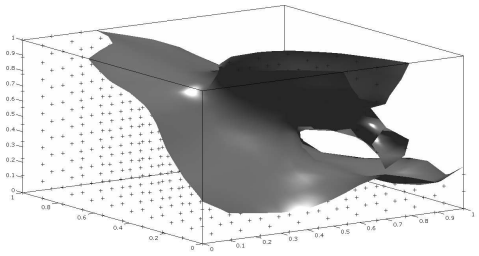
(a) Example 1 in dimension 2



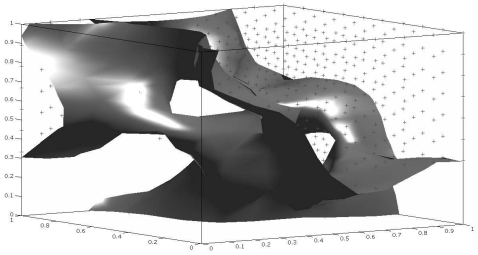
(b) Example 2 in dimension 2



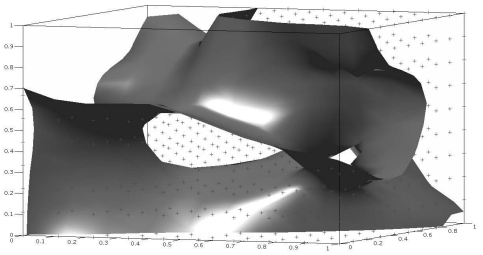
(c) Example 1 in dimension 3



(d) Example 2 in dimension 3



(e) Example 3 in dimension 3



(f) Example 4 in dimension 3

Figure 6: Examples of hard learning problems in dimension two and three

4.3 How to generate a low dispersion sequence

In this section, we review algorithms to generate low dispersion sequences and present our new algorithm. We recall that dispersion of a sequence x is defined by: $\delta(x) = \sup_{y \in I^s} \min_{i=1,\dots,n} d(y, x_i)$.

Yakowitz et al. (2000) says that, *in dimension $s \geq 2$, for every n, generate an n-sequence in the unit-cube which minimizes dispersion is a hard unresolved problem and the optimal value of it is also unknown.* Lindemann and LaValle (2004) and LaValle and Branickly (2004) have developed an incremental algorithm reducing the dispersion in the particular case of applications in robotics based on trees, getting around this problem of minimization. In space filing design literature, it is often written that samples minimizing dispersion, also called *maximin* samples, are very hard to obtain, all the more when dimension is high.
However, problem of generating the point sets with the lowest dispersion for the s dimensional unit torus is solved and solution are the Voronoï¯'s principal lattices of

Dimension	Sample size	Learning method	Grid		LDS1		LDS2	
			value	*rank*	value	*rank*	value	*rank*
2	100	SVM	**3.85**	*1*	4.08	*2*	4.29	*3*
		KNN	**3.81**	*1*	4.75	*2*	5.26	*3*
	225	SVM	**1.30**	*1*	1.33	*2*	1.36	*3*
		KNN	**1.50**	*1*	2.00	*3*	2.06	*2*
	400	SVM	**0.96**	*1*	0.99	*2*	1.02	*3*
		KNN	**1.10**	*1*	1.55	*2*	1.66	*3*
	676	SVM	**1.06**	*1*	1.06	*2*	1.08	*3*
		KNN	**1.11**	*1*	1.55	*2*	1.60	*3*
	2500	SVM	**0.37**	*1*	0.39	*3*	0.38	*2*
		KNN	**0.43**	*1*	0.60	*2*	0.71	*3*
3	64	SVM	**5.6**	*1*	5.6	*2*	5.8	*3*
		KNN	**6.5**	*1*	8.0	*3*	7.7	*2*
	512	SVM	**2.19**	*1*	2.25	*3*	2.20	*2*
		KNN	**3.05**	*1*	3.87	*3*	3.70	*2*
	1000	SVM	**1.06**	*1*	1.06	*1*	1.08	*2*
		KNN	**1.12**	*1*	1.55	*3*	1.60	*2*
	3375	SVM	2.49	*2*	2.47	*1*	**2.46**	*1*
		KNN	**2.34**	*1*	3.15	*3*	2.97	*2*
4	2401	SVM	**1.57**	*1*	1.56	*2*	1.57	*3*
		KNN	**3.40**	*1*	3.90	*2*	3.91	*3*
	4096	SVM	1.33	*1*	1.33	*1*	**1.32**	*1*
		KNN	**2.93**	*1*	3.41	*3*	3.47	*2*
	6561	SVM	1.14	*1*	**1.11**	*1*	**1.11**	*1*
		KNN	**2.50**	*1*	2.97	*2*	2.96	*2*
	10000	SVM	2.93	*3*	**2.84**	*1*	2.90	*2*
		KNN	**4.93**	*1*	5.41	*2*	5.55	*3*

Table 1: Average and rank of generalization error on 1000 simple classification problems (in %). For each set of problems and learning method, the best results are highlighted in bold.

the first type (Yakowitz et al., 2000). Generate a low dispersion sequence in unit torus is easier than in unit cube because torus doesn't have (oriented) vertex. An alternative consists in maximizing the criterion $\delta_2(x) = \inf_{(x_1,x_2)\in I^{s2}} d(x_1, x_2)$ instead of minimizing $\sup_{y\in I^s} \min_{i=1,\ldots,n} d(y, x_i)$ because it is easier to optimize numerically. Nevertheless this optimization pushes points towards the frontier of cube. This effect can be avoided by considering the criterion $\delta_3(x) = \inf_{(x_1,x_2)\in I^{s2}} d(x_1, \{x_2\} \cup I^{s'})$ where $I^{s'} = \{x \in \mathbb{R}^s | x \notin I^s\}$. A set which maximizes $\delta_3(x)$ is said a *minimax set*. Teytaud et al. (2007) also have developed an algorithm based on criterion *minimax* and trees.

We propose an algorithm to generate a low dispersion sequence based on spring variables, function of distance between points. Although it was not originally designed for its computational efficiency, this algorithm has good proprieties in prac-

tice: for an appropriate number of points, it converge to the Shukarrev grid which minimizes dispersion and it converges generally quickly. Its complexity is about $O(n^2 s)$ for a sequence of size $n$and s iterations (experimentally ten iterations are usually enough).

Algorithm: From a sequence S with n points, $S = \{x_i\}_{i=1,\dots,n}$ in dimension s, we spread points in order to decrease dispersion inspired by k-near neighbors. First, we compute a distance limit beyond which a point does not influence others. For each point x of S, we only consider neighbors x_i of S with distance inferior to a distance limit $distance_{threshold}$. We compute so a spring variable between x and each neighbor x_i defined by $\frac{2 * distance_{threshold} - d(x, x_i)}{2 * distance_{threshold}}$. When points are very far away, this variable is near of $1/2$ and when points are near, it is nearest one. Then, we space point x proportionately to each spring variable into the opposite direction at each point x_i: the closer the points are, the more the algorithm spaces them. After each iteration of the algorithm, we compute the number of point on each edge of I^s. If there are more than $\lceil \sqrt[s]{n} \rceil$ points, we stochastically remove one of them and put it stochastically on the middle of I^s. With this action, we limit the number of point near each edge. So we realize a local dispersion minimization which becomes global after iterating this process. Stopping criterion can be either a maximal number of iteration, or the stabilization of the dispersion.

Algorithm 1 Generation of a low dispersion sequence

threshold $\leftarrow \frac{1}{\sqrt[s]{n}}$
$S2 \leftarrow S$
repeat
  for all $x \in S$ **do**
    $move \leftarrow 0$
    for all $y \in S | y \neq x$ **do**
      if $d(x, y) < threshold$ **then**
        $t = \frac{2 * threshold - d(x,y)}{2 * threshold}$
        $move = move + t * (x - y)$ {compute neighbor's influence}
      end if
      $S2(x) \leftarrow S(x) + move$
    end for
    $S \leftarrow S2$
    for all $x \in S$ **do**
      for $i = 1$ to s **do**
        $x_i = \min\left(1 - \frac{threshold}{4}, \max\left(x_i, \frac{threshold}{4}\right)\right)$ {Prevent points from crossing the border}
      end for
    end for
  end for
until Stop criterion attained

Demonstration of our algorithm in dimension 2 can be see at http://www.jmlr.org/papers/ .

4.4 Experiments about dispersion

In this section, we study experimentally classification learning performence with dispersion of samples. At first, we complete experiments of section 4.2. We will see that learning performance is clearly linked to dispersion of samples. In a second part, we will prove that minimizing dispersion of samples is a relevant strategy to minimize learning error.

We have seen in previous section that minimizing discrepancy is not an appropriate strategy. We propose to show you experimentally that dispersion has a beneficial effect on generalisation error. Using the same experiments as in section 4.2, we have computed dispersion as samples. Table 4.4 shows these values.

Dimension	Sample size	Grid	LDS1	LDS2
2	100	**7.9**	12.7	13.8
	225	**5.0**	8.4	8.9
	400	**3.7**	7.5	8.9
	676	**2.8**	4.9	5.2
	2500	**1.4**	2.9	2.4
3	64	**28.9**	32.4	36.3
	512	**12.4**	15.5	18.3
	1000	**9.6**	14.0	14.9
	3375	**6.2**	10.7	8.9
4	2401	**16.7**	21.0	25.2
	4096	**14.3**	16.4	19.2
	6561	**12.5**	16.4	17.8
	10000	**11.1**	14.6	15.3

Table 2: Dispersion of learning samples used in numerical experiments of section 4.2 ($*10^{-2}$). Best dispersions are highlighted in bold: it is always dispersion of grid.

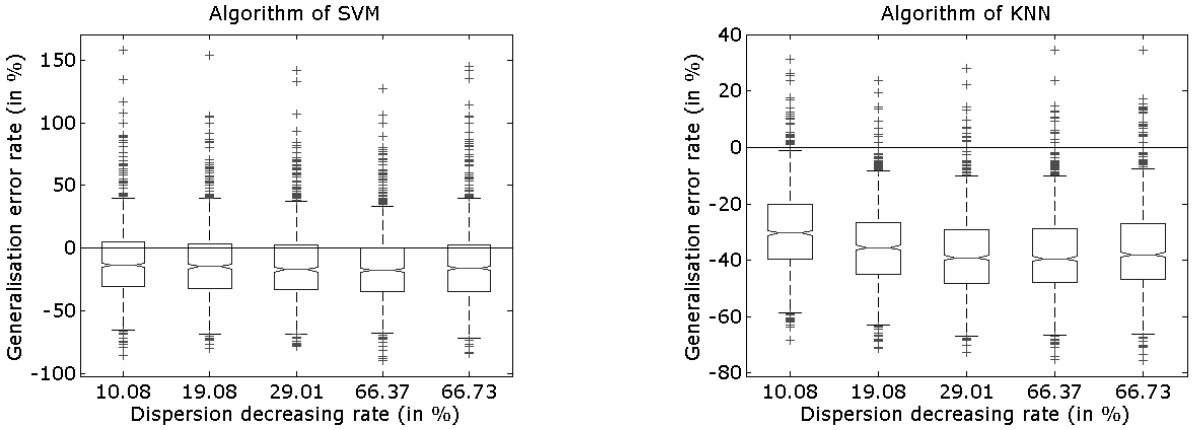
Comparing table 4.2 to table 4.4, we can remark that regular grid, which has best results on generalisation error, has as well the lower dispersion. For a fixed dimension, the more less is the sample size, the more it is obvious. Moreover we can also see that LDS1 has a lower dispersion than LDS2 [7] and has a rank which is often in front of LDS2 rank. As a consequence, it supposes that minimizing dispersion is the relevant criterion to optimize learning sample.

In order to prove it more categorically, we have made another type of experiments. The first set of them has consisted of generating a learning set of fixed size,

7. Except for samples of size 2500 in dimension 2 and size 3375 in dimension 3.

learning 1000 classification problems and estimating generalisation error for each problem.

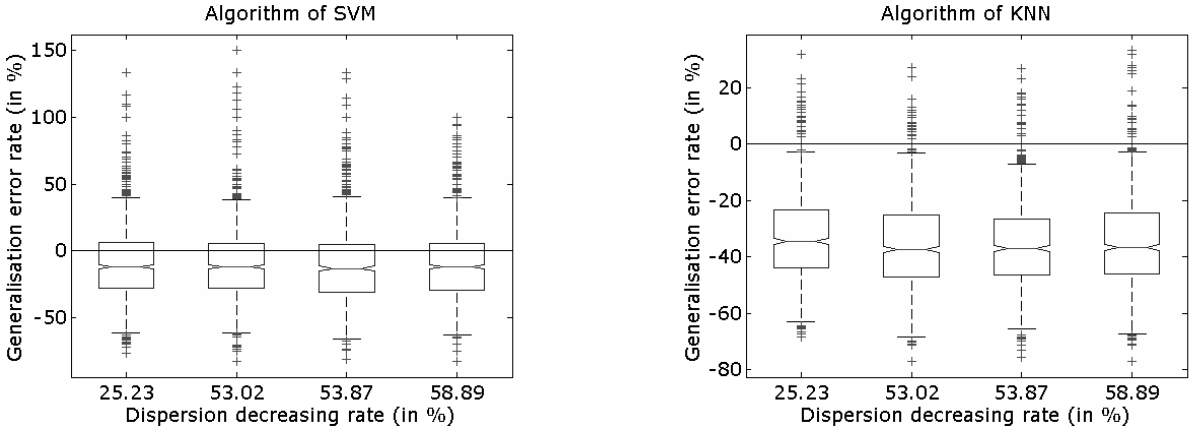
We have used the same protocol for learning as in section 4.2 except for estimating parameter σ of SVM which is estimated for each problem at the first step on the test sample. Then, with algorithm describted in section 4.3, we have moved points of sample in order to decrease their dispersion. We have one after the other made learning for the previous classification problems and have of course estimated generalisation error. We have repeat this process some numbers of times. In the last step, we have studied distribution of evolution of generalisation error rate functions of dispersion decreasing rate.



(a) Example 1 in dimension 2

(b) Example 2 in dimension 2

Figure 7: Results for simple learning problems in dimension two



(a) Example 1 in dimension 2

(b) Example 2 in dimension 2

Figure 8: Results for hard learning problems in dimension two

5 Conclusion and discussion

Cervellera and Muselli (2004) and Mary (2005) propose discrepancy as criterion to generate sample for regression learning problems. Based their work, we tried to

transfer their results to classification problems without success and we have experimentally proved that minimizing discrepancy is not a good strategy for classification.

We proposed another criterion for optimizing samples for classification : the dispersion. We have established a theoretical linear link between dispersion and generalization error in classification within the context of a simple learning algorithm. Moreover, we have realized experimental tests on SVMs and KNN which show a link between dispersion and generalization error.
We have not compared low discrepancy samples with uniform random samples. However Iwata and Ishii (2002) have observed experimentally a gain of quality in classification with the multilayer perceptron using low discrepancy samples instead of random samples. It does not contradict our results. Indeed dispersion of a random sample is generally higher than dispersion of low discrepancy sample. As a consequence, learning with low dispersion samples is better than learning with low discrepancy samples, which is also better than learning with uniform random samples for classification problems.

At first sight, difference of learning behavior between classification and regression towards learning samples was very surprising. To complete, we want to provide the reader with some insight on this particularity. Equation 2 explains clearly that learning error is directly proportional to two independent terms: discrepancy of learning sample on the one hand, and variation of Hardy-Krause of a function on the other hand.
In the setting of regression, this variation is the sum on all possible combinations of space directions of terms, these terms being the characterization of the derivative function evolution. Hence, it is essential, in order to catch correctly this variation, to have an information on the function which has to be well distributed on all the space (we don't want to have area without information) and well distributed according to all possible directions. As a consequence it is necessary to avoid regular structure within information. Low discrepancy sequences, which have good properties of projection on each axes, are then a good compromise solution for good repartition and absence of structure.
In the case of classification, variation does not play any role : it is equal to infinity (excepted for very particular cases). So it is very important to have a local information at each point of space. In other words, we want to minimize area without local information: it is equivalent to minimizing dispersion of samples !

To conclude, it is interesting to generate data with low dispersion for classification problem or for approximating boundaries of surfaces, especially as number of data is limited. A good strategy consists in using an iterative learning scheme as proposed by Chapel and Deffuant (2006): the first step consists in learning classification boundaries with a fixed sized and low dispersion data set. Then we generate data with a higher density and small dispersion near the boundary of classification function detected at the previous step. This could enhance significantly the learning performance obtained with fixed size but freely chosen sample for a classification problems. and could then limit costly data harvesting or computer experiments in different applications.

Appendix A.

In this appendix we present mathematically the notion of variation of a function and explain which functions have a finite variation in the sense of Hardy-Krause:

Definition of the variation of a function

Let a function $\varphi : I^s \to \mathbb{R}$ which we want to compute the variation.
For each vertex of a given subinterval $B = \Pi_{i=1}^{s}[a_i, b_i]$ of I^s, we can define a binary label by assigning '0' to every a_i and '1' to every b_i.
We define $\Delta(\varphi, B)$ as the alternating sum of computed at the vertices of B, this is $\Delta(\varphi, B) = \sum_{x \in e_B} \varphi(x) - \sum_{x \in o_B} \varphi(x)$ where e_B is the set of vertices with an even number of '1's in their label and o_B is the set of vertices with an odd number of '1's in their label. Denoting $\mathcal{P}$ a partition of I^s into subintervals.
The variation of φ on I^s in the sense of Vitali is defined by:

$$V^{(s)}(\varphi) = \sup_{\mathcal{P}} \sum_{B \in \mathcal{P}} |\Delta(\varphi, B)| .$$

For $1 \leq k \leq s$ and $1 \leq i_1 < i_2 < \cdots < i_k \leq s$, let $V^{(k)}(\varphi, i_1, \ldots, i_k)$ be the variation in the sense of Vitali of the restriction of φ to the k-dimensional face $\{(x_1, \ldots, x_s) \in I^s : x_i = 1 \text{ for } i \neq i_1, \ldots, i_k\}$ where x_i is the i-th component of x.
The variation of φ on I^s in the sense of Hardy and Krause is defined by:

$$V_{HK}(\varphi) = \sum_{k=1}^{s} \sum_{1 \leq i_1 < i_2 < \cdots < i_k \leq s} V^{(k)}(\varphi, i_1, \ldots, i_k)$$

In general, compute the variation of a function can be very difficult task. However, in case of functions having continuous derivatives, the computation of upper bounds for the variation can be much simpler. If the partial derivatives of φ are continuous on I^s, it is possible to write the variation in the sense of Vitali $V^{(s)}(\varphi)$ in an easier way as:

$$V^{(s)}(\varphi) = \int_0^1 \ldots \int_0^1 \left| \frac{\partial^s \varphi}{\partial x_1 \ldots x_s} \right| dx_1 \ldots dx_s.$$

How to ensure a bounded variation

To be with a bounded variation is a very strong, non-natural assumption. For example, if φ_1 and φ_2 are two linear functions on the s dimensional cube, then $min(f, g)$ has a finite variation in the sense of Hardy-Krause when $s = 2$, but it is not necessarily so when $s > 2$. An other example is the case of the indicator functions. An indicator function in 2 dimension must either have positive variation in Vitali's sense, or must have at least one input variable on which it does not truly depend. The same is not true for $s > 3$.
Functions which have all its continuous derivatives and these derivatives are all bounded have a finite variation : it is equivalent to be $\mathcal{C}^s$.

References

C. Cervellera and M. Muselli. A Deterministic Learning Approach Based on Discrepancy. In *Proceedings of WIRN'03*, pages 53–60, 2003.

C. Cervellera and M. Muselli. Deterministic Design for Neural Network Learning: An Approach Based on Discrepancy. In *Proceedings IEEE Transactions on Neural Network*, volume 15, pages 533–544, 2004.

C. Cervellera, M. Maccio, and M. Muselli. Deterministic Learning for Maximum-Likelihood Estimation Through Neural Network Learning. In *Proceedings IEEE Transactions on Neural Network*, volume 19, pages 1456–1467, 2008.

C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

L. Chapel and G. Deffuant. SVM viability controller active learning. In *Kernel machines for reinforcement learning workshop, Pittsburgh, PA*, 2006.

L. Chapel and G. Deffuant. SVM approximation of value function contours in target hitting problems. *Automatica*, submitted.

G. Deffuant, L. Chapel, and S. Martin. Approximating viability kernels with support vector machines. *In IEEE transactions on automatic control,*, 52(5):933–937, 2007.

M. Drmota and R.F. Tichy. *Sequences, Discrepancies and Applications*. Springer, 1997.

K. Iwata and N. Ishii. Discrepancy as a Quality Measure for Advoiding Classification Bias. In *Proceedings of the 2002 IEEE International Symposium on Intelligent Control. Vancouver, Canada.*, 2002.

A. Jourdan and I. Zabalza-Mezghani. Response Surface Designs for Scenario Management and Uncertainty Quantification in Reservoir Production. *Mathematical Geology*, 36(8):965 – 985, 2004.

S.M. LaValle and S.R. Branickly. On the Relationship Between Classical Grid Search and Probabilistic Roadmaps. *International Journal of Robotics Research*, 2004.

S.R. Lindemann and S.M. LaValle. Incrementally Reducing Dispersion by Increasing Voronoi Bias in RRTs. In *IEEE International Conference on Robotics and Automation*, 2004.

J. Mary. *Etude de l'Apprentissage Actif, Application la Conduite d'Expriences*. PhD thesis, Universit Paris XI, 2005.

W.J. Morokoff and R.E. Caflisch. Quasi-Monte Carlo Integration. *J. Comput. Phys.*, 122(2):218–230, 1995.

H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics, 1992.

A.B. Owen. Multidimensional Variation for Quasi-Monte Carlo. 2004. `www-stat.stanford.edu/~owen/reports/ktfang.pdf`.

W.H. Press and S.A. Teukolsky. Quasi- (that is, sub-) Random Numbers. *Computers in Physics*, 3(6):76 – 79, 1989.

B. Schlkpf and A.J. Smola. *Learning with Kernels: Support Vector Machines, Regularisation, Optimization, and Beyond*. The MIT Press, 2002.

O. Teytaud, S. Gelly, and J. Mary. Active learning in regression, with application to stochastic dynamic programming. In *Proceedings of International Conference on Informatics to Control, Automation and Robotics*, 2007.

S. Tezuka. Uniform Random Numbers: Theory and Practice. *The Kluwer International Series in Engineering and Computer Science Mathematics*, 1995.

E. Thiedmard. An Algorithm to Compute Bounds for the Star Discrepancy. *Journal of Complexity*, 17(4):850 – 880, 2001.

V.N. Vapnick. *The Nature of Statistical Learning*. Springer-Verlag, 1995.

V.N. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Prob. Its Applic*, 16:264–280, 1971.

S. Yakowitz, P. L'Ecuyer, and F. Vazquez-Abad. Global stochastic optimization with low-dispersion point sets. *Operations Research*, 48:939–950, 2000.